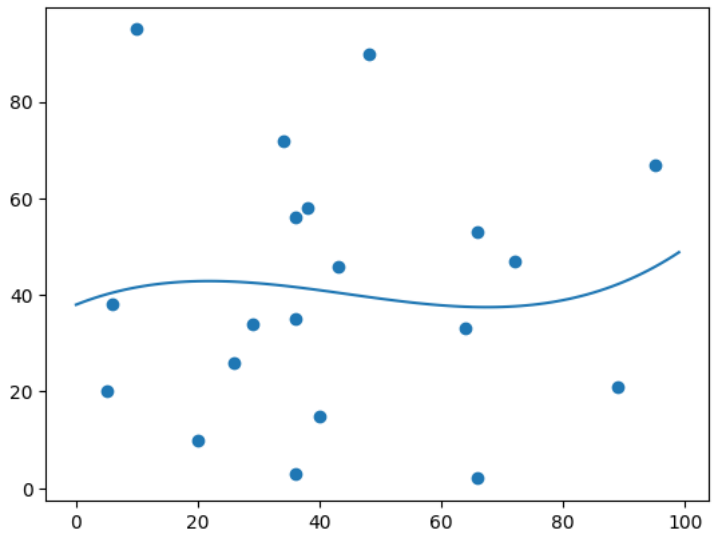
Polynomial_regression**

```
import numpy as np
import matplotlib.pyplot as plt
x=[89,43,36,36,95,10,66,34,38,20,26,29,48,64,6,5,36,66,72,40]
y=[21,46,3,35,67,95,53,72,58,10,26,34,90,33,38,20,56,2,47,15]
model = np.poly1d(np.polyfit(x,y,3)) # 3 degree curve
myline = np.linspace(1,95,100) # 100 is showing no of sample point
plt.scatter(x,y)
plt.plot(model(myline))
plt.show()
```

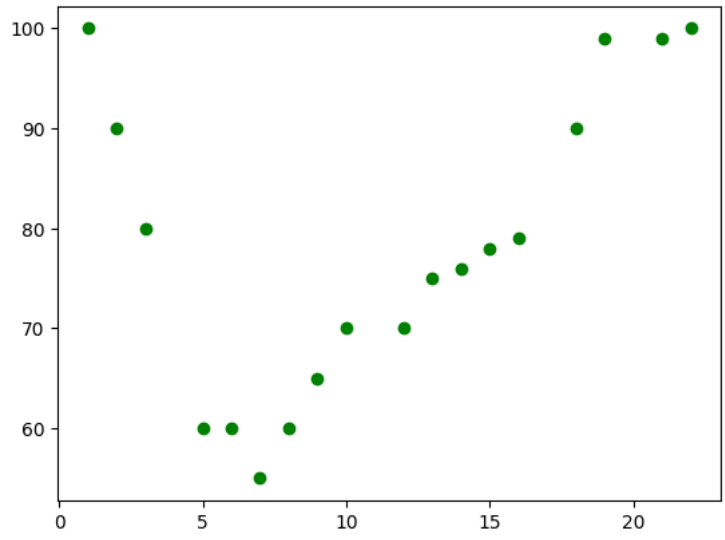



```
# R square value
from sklearn.metrics import r2_score
print(r2_score(y,model(x)))
```

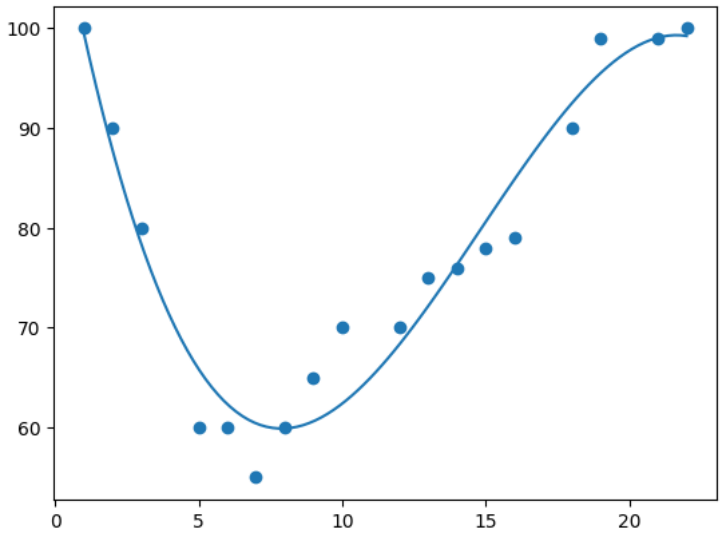
```
0.009952707566680652
```

Best Fit

```
import matplotlib.pyplot as plt
x = [1,2,3,5,6,7,8,9,10,12,13,14,15,16,18,19,21,22]
y = [100,90,80,60,60,55,60,65,70,70,75,76,78,79,90,99,99,100]
plt.scatter(x,y, color ="green")
plt.show()
```



```
model = np.poly1d(np.polyfit(x,y,3)) # 3 degree curve
myline = np.linspace(1,22,100) # 100 is no of sample points showing
plt.scatter(x,y)
plt.plot(myline, model(myline))
plt.show()
```




```
from sklearn.metrics import r2_score
print(r2_score(y,model(x)))
```

```
0.9432150416451026
```

```
model = np.poly1d(np.polyfit(x,y,3))
pred = model(1)
print(pred)
```

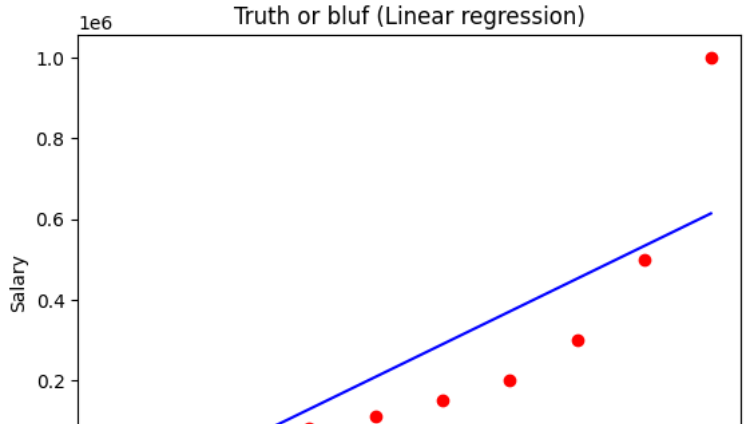
```
99.54274392967326
```

Hands on Examples

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
dataset = pd.read_csv('https://s3.us-west-2.amazonaws.com/public.gamelab.fun/dataset/position_salaries.csv')
X= dataset.iloc[:,1:2].values
y= dataset.iloc[:,2].values
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2,random_state=0)
```

```
# fitting linear regression to dataset
from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression().fit(X,y)
# Visualizing the linear regression model result
def viz_linear():
    plt.scatter(X,y,color="red")
    plt.plot(X,lin_reg.predict(X),color="blue")
    plt.title("Truth or bluf (Linear regression)")
    plt.xlabel("Position level")
    plt.ylabel("Salary")
    plt.show()
    return
viz_linear()
```




```
# Predicting a new result with linear regression
pred_linear = lin_reg.predict([[11]])
```

Support_vector_Machines

```
from sklearn import datasets
# Load dataset
cancer = datasets.load_breast_cancer()
```

```
# print the name of 30 features
print("Features:",cancer.feature_names)
# print the label type of cancer
print("labels:",cancer.target_names)
```

```
Features: ['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
labels: ['malignant' 'benign']
```

```
# print data(feature) shape
cancer.data.shape
```

```
(569, 30)
```

```
# PRINT THE cancer data feature(top 5 records)
print(cancer.data[0:5])
```

```
[[1.799e+01 1.038e+01 1.228e+02 1.001e+03 1.184e-01 2.776e-01 3.001e-01
  1.471e-01 2.419e-01 7.871e-02 1.095e+00 9.053e-01 8.589e+00 1.534e+02
  6.399e-03 4.904e-02 5.373e-02 1.587e-02 3.003e-02 6.193e-03 2.538e+01
  1.733e+01 1.846e+02 2.019e+03 1.622e-01 6.656e-01 7.119e-01 2.654e-01
  4.601e-01 1.189e-01]
 [2.057e+01 1.777e+01 1.329e+02 1.326e+03 8.474e-02 7.864e-02 8.690e-02
  7.017e-02 1.812e-01 5.667e-02 5.435e-01 7.339e-01 3.398e+00 7.408e+01
  5.225e-03 1.308e-02 1.860e-02 1.340e-02 1.389e-02 3.532e-03 2.499e+01
  2.341e+01 1.588e+02 1.956e+03 1.238e-01 1.866e-01 2.416e-01 1.860e-01
  2.750e-01 8.902e-02]
 [1.969e+01 2.125e+01 1.300e+02 1.203e+03 1.096e-01 1.599e-01 1.974e-01
  1.279e-01 2.069e-01 5.999e-02 7.456e-01 7.869e-01 4.585e+00 9.403e+01
  6.150e-03 4.006e-02 3.832e-02 2.058e-02 2.250e-02 4.571e-03 2.357e+01
  2.553e+01 1.525e+02 1.709e+03 1.444e-01 4.245e-01 4.504e-01 2.430e-01
  3.613e-01 8.758e-02]
 [1.142e+01 2.038e+01 7.758e+01 3.861e+02 1.425e-01 2.839e-01 2.414e-01
  1.052e-01 2.597e-01 9.744e-02 4.956e-01 1.156e+00 3.445e+00 2.723e+01
  9.110e-03 7.458e-02 5.661e-02 1.867e-02 5.963e-02 9.208e-03 1.491e+01
  2.650e+01 9.887e+01 5.677e+02 2.098e-01 8.663e-01 6.869e-01 2.575e-01
  6.638e-01 1.730e-01]
 [2.029e+01 1.434e+01 1.351e+02 1.297e+03 1.003e-01 1.328e-01 1.980e-01
  1.043e-01 1.809e-01 5.883e-02 7.572e-01 7.813e-01 5.438e+00 9.444e+01
  1.149e-02 2.461e-02 5.688e-02 1.885e-02 1.756e-02 5.115e-03 2.254e+01
  1.667e+01 1.522e+02 1.575e+03 1.374e-01 2.050e-01 4.000e-01 1.625e-01
  2.364e-01 7.678e-02]]
```

```
# print the cancer labels 0:malingnant 1: benign
print(cancer.target)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1 0 1 0 0 1 1 1 1 0 1 0 0
 1 0 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 1 1 1 0 1 1 0 1 1
 1 1 1 1 1 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 1 1 0 1 1 0 1 1 1 1 0 1
 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1 1 0 0 1 1 0 0 1 1 1 1 0 1 1 0 0 0 1 0
 1 0 1 1 1 0 1 1 0 0 1 0 0 0 0 1 0 0 0 1 0 1 0 1 1 0 1 0 0 0 0 1 1 0 0 1 1
 1 0 1 1 1 1 1 1 0 0 1 1 0 1 1 0 0 1 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 0 1 1 0 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1
 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 0 1 1 1 1 0 0 0 1 1
 1 1 0 1 0 1 0 1 1 1 0 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0
 0 1 0 0 1 1 1 1 1 0 1 1 1 1 1 1 0 1 1 1 0 1 1 0 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1
 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 1 1 1 1 1 0 1 1
 0 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1
 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 0 0 1 0 1 0 1 1 1 1 1 0 1 1 0 1 0 1 0 0
 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 0 0 0 0 0 0 1]
```

```
# import train test split function
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(cancer.data,cancer.target,test_size=0.2,random_state=0)
```

```
# import svm model
from sklearn import svm
# Create a SVM Classifier
clf = svm.SVC(kernel='linear') # linear kernal
# train the model using training set
clf.fit(X_train, y_train)# predict the response of the test dataset
y_pred = clf.predict(X_test)
# predict the response of the test dataset
y_pred = clf.predict(X_test)
```

```
# import sklearn metrics module for accuracy calculation
from sklearn import metrics
score = metrics.accuracy_score(y_test, y_pred)
print("Accuracy:", score)
```

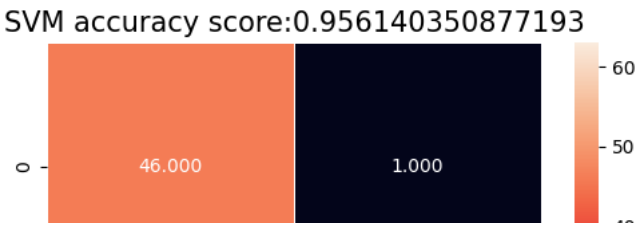
```
Accuracy: 0.956140350877193
```

```
# confusion matrix
from sklearn import metrics
cm = metrics.confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[46  1]
 [ 4 63]]
```

```
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure("figsize"== (12,12))
sns.heatmap(cm, annot=True, fmt=".3f", linewidths=.5, square = True)
plt.xlabel('Predicted Output')
plt.ylabel('Actual Output')
all_sample_title = 'SVM accuracy score:{0}'.format(score)
plt.title(all_sample_title, size = 15)
```

```
Text(0.5, 1.0, 'SVM accuracy score:0.956140350877193')
```



K_Nearest_neighbours

```
import pandas as pd
df = pd.read_csv("mldata1.csv")
df["gender"] = df["gender"].replace("Male",1)
df["gender"] = df["gender"].replace("Female",0)
df.head()
```

	age	height	weight	gender	likeness
0	27	170.688	76.0	1	Biryani
1	41	165	70.0	1	Biryani
2	29	171	80.0	1	Biryani
3	27	173	102.0	1	Biryani
4	29	164	67.0	1	Biryani

```
# selection of input and output variable
X = df[["weight","gender"]]
y = df["likeness"]
```

```
# Machine learning algorithm
from sklearn.neighbors import KNeighborsClassifier
# Create and fit our model
model = KNeighborsClassifier(n_neighbors=9)
model.fit(X,y)
# predict the result
predicted =model.predict([[59,1]]) # 70 Weight, 1 Male
predicted
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but KNeighborsClassi
  warnings.warn(
array(['Biryani'], dtype=object)
```

```
# Split data into test and train(80/20)
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2)
1
#Create and fit a model
model = KNeighborsClassifier(n_neighbors=9).fit(X_train,y_train)
# predicting output
predicted_values = model.predict(X_test)
predicted_values
```

```
array(['Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Samosa',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Samosa',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Samosa', 'Biryani', 'Biryani',
       'Biryani', 'Biryani', 'Biryani', 'Biryani', 'Samosa', 'Biryani',
       'Biryani'], dtype=object)
```

```
# checking score
score = accuracy_score(y_test, predicted_values)
score
```

```
0.6122448979591837
```

✓ 0s completed at 6:43 AM